SIAG/OPT Views-and-News

A Forum for the SIAM Activity Group on Optimization

Volume 11 Number 2 August 2000

Contents

Case Studies

Optimization of Engineering Systems Governed by Differential Equations: Editor's Introduction

Natalia Alexandrov
Multidisciplinary Optimization Branch
Aerospace Systems Concepts & Analysis
NASA Langley Research Center, Hampton, Virginia
23681-2199, USA n.alexandrov@larc.nasa.gov

Many engineering optimization problems fall into one of three categories: optimal design, parameter estimation, and optimal control. Their common feature is that differential equations appear in the problem formulation as conditions that must be satisfied for a physical system to be realizable or constructible. This issue of V&N is devoted to some aspects of problem solving in engineering optimization. A brief discussion of the multidisciplinary optimization (or optimal design) problem will serve to introduce some terminology to the readers unfamiliar with this area and to place the subjects of the invited articles in the context of several active research directions.

Multidisciplinary optimization (MDO) may be viewed as a collection of systematic approaches to the design optimization of complex, coupled engineering systems (e.g., [1]), where "multidisciplinary" refers to different aspects of a design problem. For instance, the design of aerospace vehicles involves aerodynamics, structural analysis, propulsion, and control, among many other disciplines.

Because the total design problem is an amorphous creature, not easily quantified or even defined (for reasons both sociological and technical), researchers who develop methods for optimal design, find it convenient to abstract the MDO problem as a subset of the total design problem that can be formulated as the following (here simplified) nonlinear program:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x, u(x)) \\ \text{subject to} & h(x, u(x)) = 0 \\ & g(x, u(x)) \geq 0, \end{array} \tag{1}$$

where x is the vector of design variables and $u(x)$ is defined via a block system of equations,

$$A(x, u(x)) = \begin{pmatrix} A_1(x, u_1(x), \ldots, u_N(x)) \\ \vdots \\ A_N(x, u_1(x), \ldots, u_N(x)), \end{pmatrix} = 0 \tag{2}$$

for N blocks. The blocks represent the state equations of the disciplinary *analyses* (known as *simulations* if they are accurate and expensive) and the interdisciplinary couplings. The state equations normally form a set of coupled differential equations that describe the behavior of the disciplinary subsystems. Given the *design variables* x, a disciplinary analysis $A_i((x, u_1(x), \ldots, u_N(x)) = 0$ solves for the *state variables* or disciplinary *responses* $u_i(x)$. Solving (2) leads to a full multidisciplinary analysis, or MDA, in which the coupled disciplines give a physically consistent—and thus meaningful—result. For instance, in aerospace MDO, a two-discipline problem might represent the aeroelastic interaction between aerodynamics (A_1) and structural analysis (A_2) for a wing in steady-state flow. Then u_1 and u_2 may represent the flow field near the wing and the deformed shape of the wing due to structural response and aerodynamic loads, respectively. Computing the flow field u_1 requires the shape of the wing contained in u_2, while computing the wing deformation u_2 requires the aerodynamic loads, contained in u_1.

Traditionally, at each iteration of optimization, the design variable vector x is passed to the MDA system. The coupled PDEs are then solved for the state vector u, thus reducing the dimension of (1) by making it a problem in x only. Optimizers will recognize this method as a *variable reduction* or reduced-space approach.

Among numerous difficulties associated with solving (1), the most pronounced one has to do with the frequently prohibitive expense of repeated evaluation of functions and derivatives via simulations. Several current interrelated research directions aim to overcome this difficulty.

The expense of implementing and executing a straightforward, conventional optimization approach to (1) has—for a long time—motivated researchers to propose alternative optimal design problem formulations and attendant optimization algorithms. Historical notes and recent efforts in analysis and development of problem formulations for MDO can be found in [2, 3, 4] and references therein.

Simultaneous Analysis and Design (SAD or SAND) (e.g., [5, 6]) is an example of an alternative problem formulation. It is motivated by the experience which suggests that allowing infeasibility with respect to the MDA manifold far from solutions may reduce the cost of optimization. SAND treats both the design variables and the state variables as explicit optimization variables, while (2) is treated as explicit equality constraints. The article by Omar Ghattas and George Biros in this issue discusses promising recent advances in using optimization methods for solving reduced-space and full-space (SAND) formulations.

Affordable computation of derivatives for any engineering design problem is an important component of an optimization procedure. The "adjoint approach" is frequently used for computing derivatives in problems governed by PDE. Michael Lewis explains the adjoint approach in this issue for readers unfamiliar with the technique. Interesting recent results in computing derivatives for optimal design problems can also be found in, e.g., [7, 8] and references therein.

Another general research direction addresses the expense of using simulations in computational optimization by developing adaptive modeling techniques and optimization strategies for a systematic use of variable-fidelity models.

Models and approximations of varying degree of fidelity have been used in engineering for a long time (e.g., see [9] for a review in the area of structural optimization), with optimization procedures largely based on heuristics. Due to improvements in the numerical modeling techniques and the increased availability of high-fidelity analyses, optimization with variable-fidelity models and approximations has become a subject of much interest in the past few years (e.g., [10, 11, 12]).

A variety of interesting modeling techniques—such as reduced-order models [13, 14] and space mapping methods [15]—have been under study in a number of engineering areas. In this issue, Tony Patera and co-authors describe some of the exciting recent developments in reduced-order modeling at MIT.

Variable-fidelity models must be managed in optimization procedures in ways that minimize the expense, yet guarantee convergence to high-fidelity answers. Recent model management methods include techniques that use sensitivities (e.g., [16, 17]) and techniques that do not (e.g., [18]). The latest results in model management with variable-resolution models [19] and variable physical fidelity models [20] are

promising.

Optimization of engineering systems involves many other areas of research, such as computer science (for computational frameworks and problem integration), and other areas of optimization, such as optimization under uncertainty. Interested readers may find papers from a variety of contributing fields in [21]. Last-minute news in some of the areas described in this issue may be heard at the minisymposia titled "Optimization of Engineering Systems Governed by Differential Equations", Parts I and II, at the First SIAM Conference on Computational Science and Engineering, Sept. 21–24, Washington DC. Finally, all optimizers are invited to contribute to the new journal "Optimization and Engineering" published by Kluwer[1].

REFERENCES

[1] J. Sobieszczanski-Sobieski and R. T. Haftka. Multidisciplinary aerospace design optimization: Survey of recent developments. *Structural Optimization*, 14(1):1–23, August 1997.

[2] N. M. Alexandrov and R. Michael Lewis. Analytical and computational aspects of collaborative optimization. National Aeronautics and Space Administration, NASA/TM–210104–2000, April 2000.

[3] —. Analytical and computational properties of distributed approaches to MDO. AIAA-2000-4718, 2000.

[4] —. Algorithmic perspectives on problem formulations in MDO. AIAA-2000-4719, 2000.

[5] R. T. Haftka, Z. Gürdal, and M. P. Kamat. *Elements of structural optimization*. Kluwer Academic Publishers, Dordrecht, 1990.

[6] R. M. Lewis. Practical aspects of variable reduction formulations and reduced basis algorithms in multidisciplinary design optimization. In N.M. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State-of-the-Art*, Philadelphia, 1997. SIAM.

[7] A. A. Giunta. Sensitivity analysis for coupled aerostructural systems. NASA-TM-1999-209367, August 1999.

[8] J. Borggaard and J. A. Burns. Asymptotically consistent gradients in optimal design. In N. M. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art*, pages 303–314. SIAM, 1997.

[9] J.-F. M. Barthelemy and R. T. Haftka. Approximation concepts for optimum structural design—a review. *Structural Optimization*, 5:129–144, 1993.

[10] S. Burgee, A.A. Giunta, V. Balabanov, B. Grossman, W.H. Mason, R. Narducci, R.T. Haftka, and L.T. Watson. A coarse-grained parallel variable-complexity multidisciplinary optimization problem. *The International Journal of Supercomputing Applications and High Performance Computing*, 10(4):269–299, Winter 1996.

[11] A.A. Giunta. *Aircraft Multidisciplinary Optimization Using Design of Experiments Theory and Response Surface Modeling Methods*. PhD thesis, Department of Aerospace and Ocean Engineering, Virginia Tech, Blacksburg, VA, 1997.

[12] J. C. Otto, M. Paraschivoiu, S. Yeşilyurt, and A. T. Patera. Bayesian-validated computer-simulation surrogates for optimization and design. In N. M. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art*, pages 368–390. SIAM, 1997.

[13] M. C. Romanowski. Reduced order unsteady aerodynamic and aeroelastic models using Karhunen-Loève eigenmodes. AIAA-96-3981-CP, 1996.

[14] W. A. Silva. Identification of linear and nonlinear aerodynamic impulse responses using digital filter techniques. NASA-TM-112872, August 1997.

[15] M. H. Bakr, J. W. Bandler, N. Georgieva, and K. Madsen. A hybrid aggressive space mapping algorithm for em optimization. *IEEE Trans. Michrowave Theory Tech*, 47, 1999.

[16] N. M. Alexandrov. A trust region framework for managing approximation models in engineering optimization. AIAA-96-4102, September 1996.

[17] R. M. Lewis. A trust region framework for managing approximation models in engineering optimization. AIAA Paper 96-4101, September 1996.

[18] A. J. Booker, J. E. Dennis, Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1–13, 1999.

[19] N.M. Alexandrov, R.M. Lewis, C.R. Gumbert, L.L. Green, and P.A. Newman. Optimization with variable-fidelity models applied to wing design. AIAA-2000-0841, January 2000.

[1] Journal information may be be found at http://fmad-www.larc.nasa.gov/mdob/users/natalia/opte/ (US site) or at http://ssor.twi.tudelft.nl/ terlaky/OPTE/opte.html (European site).

[20] N. M. Alexandrov, R. M. Lewis, and E. J. Nielsen. Aerodynamic optimization with variable-fidelity models. AIAA-2000-4886, September 2000.

[21] N. M. Alexandrov and M. Y. Hussaini, editors. *Multidisciplinary Design Optimization: State of the Art*, Philadelphia, 1997. Society for Industrial and Applied Mathematics.

Reduced-Basis Output-Bound Methods for Elliptic Partial Differential Equations[1]

Anthony T. Patera
Department of Mechanical Engineering, Room 3-264, Massachusetts Institute of Technology, Cambridge, MA, 02139-4307, patera@mit.edu
Dimitrios V. Rovas
Department of Mechanical Engineering, Room 3-243, Massachusetts Institute of Technology, Cambridge, MA, 02139-4307, rovas@mit.edu
Luc Machiels
Lawrence Livermore National Laboratory, L-561, 7000 East Avenue, Livermore, CA 94550, machiels1@llnl.gov

1. Motivation

To motivate and illustrate our methods we consider a specific example: a thermal fin. The fin, shown in Figure 1, consists of a central "post" and four "subfins;" the fin conducts heat from a prescribed uniform flux "source" at the root, Γ_{root}, through the large-surface-area subfins to surrounding flowing air. The fin is characterized by seven design parameters, or "inputs," $\mu \in \mathcal{D} \subset \mathbb{R}^{P=7}$, where $\mu^i = k^i, i = 1, \ldots, 4, \mu^5 = \text{Bi}, \mu^6 = L$, and $\mu^7 = t$. Here k^i is the thermal conductivity of the i^{th} subfin (normalized relative to the post conductivity); Bi is the Biot number, a nondimensional heat transfer coefficient reflecting convective transport to the air at the fin surfaces; and L and t are the length and thickness of the subfins (normalized relative to the post width). The performance metric, or "output," $s \in \mathbb{R}$, is chosen to be the average temperature of the fin root normalized by the prescribed heat flux into the fin root. In order to optimize the fin design, we must be able to evaluate $s(\mu)$ repeatedly and rapidly.

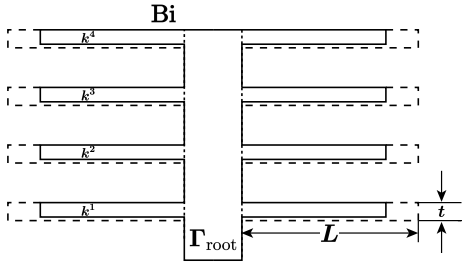

Figure 1

We can express our input-output relationship as $s = \ell^O(u(\mu))$, where $\ell^O(v)$ is a (continuous) linear functional — $\ell^O(v) = \int_{\Gamma_{\text{root}}} v$ — and $u(\mu)$ is the temperature distribution within the fin. (The temperature field is of course a function of the spatial coordinate, $\boldsymbol{x}$; we explicitly indicate this dependence only as needed.) The temperature distribution $u(\mu) \in Y$ satisfies the elliptic partial differential equation describing heat conduction in the fin,

$$a(u, v; \mu) = \ell(v), \forall v \in Y; \tag{1}$$

$a(u, v; \mu)$ is the weak form of the Laplacian, and $\ell(v)$ reflects the prescribed heat flux at the root. Here Y is the appropriate Hilbert space with associated inner product $(\cdot, \cdot)_Y$ and induced norm $\|\cdot\|_Y$ [2]. The bilinear form $a(\cdot, \cdot; \mu)$ is symmetric, $a(w, v; \mu) = a(v, w; \mu), \forall w, v \in Y^2, \forall \mu \in \mathcal{D}$; uniformly continuous, $|a(w, v; \mu)| \leq \gamma \|w\|_Y \|v\|_Y, \forall w, v \in Y^2, \forall \mu \in \mathcal{D}$; and coercive, $\alpha \|v\|_Y^2 \leq a(v, v; \mu), \forall v \in Y, \forall \mu \in \mathcal{D}$. Here α and γ are strictly positive real constants. Finally, the form $\ell(v)$ is a linear bounded functional; for our choice of scaling and output, $\ell^O(v) = \ell(v)$, which we will exploit to simplify the exposition.

It can further be shown for our problem that a can be expressed as

$$a(w, v; \mu) = \sum_{q=1}^{Q} \sigma^q(\mu) a^q(w, v), \forall w, v \in Y^2, \forall \mu \in \mathcal{D}, \tag{2}$$

[1] The material presented in this article is an expository version of work performed in collaboration with Professor Yvon Maday of University of Paris VI and reported in greater detail in references [1, 2, 3, 4]. We also thank Professor Jaime Peraire of MIT, Professor Einar Rønquist of Norwegian University of Science and Technology, Mr. Roland Von Kaenel of EPFL, and Ms. Shidrati Ali of National University of Singapore–Singapore-MIT Alliance (SMA) for helpful comments. The work is supported by AFOSR, NASA Langley Research Center, and SMA.

[2] Here $Y = H^1(\Omega)$, the space of functions that are square integrable and that have square integrable first (distributional) derivatives over the fin reference domain Ω. The inner product $(w, v)_Y$ may be chosen to be $\int_\Omega \nabla w \cdot \nabla v + wv$.

for appropriately chosen functions $\sigma^q: \mathcal{D} \to \mathbb{R}$ and associated μ-independent bilinear forms $a^q: Y \times Y \to \mathbb{R}$, $q = 1, \ldots, Q$. Note that we pose our problem on a *fixed* fin reference domain Ω in order to ensure that the parametric dependence on geometry — L and t — enters through $a(\cdot, \cdot; \mu)$ and ultimately the $\sigma^q(\mu)$. For our particular problem, $Q = 15$; if we freeze (fix) all parameters except L and t (such that $P_{\text{eff}} = 2$), $Q = 8$; if we freeze only L and t (such that $P_{\text{eff}} = 5$), $Q = 6$.

In the context of design, optimization, and control, we require very rapid response and many output evaluations. Our goal is thus to construct an approximation to $u(\mu)$, $\tilde{u}(\mu)$, and hence approximation to $s(\mu)$, $\tilde{s}(\mu) = \ell^O(\tilde{u}(\mu))$, which is (i) *certifiably* accurate, and (ii) very efficient in the limit of *many evaluations*. By the former we mean that the error in our approximate output, $|s(\mu) - \tilde{s}(\mu)|$, is *guaranteed* to be less than a prescribed tolerance ε; by the latter we mean that, following an initial *fixed* investment, the additional *incremental* cost to evaluate $\tilde{s}(\mu)$ for any new $\mu \in \mathcal{D}$ is much less than the effort required to directly compute $s(\mu) = \ell^O(u(\mu))$ by (say) standard finite element approximation.

2. Reduced-Basis Approximation

Reduced-basis methods (e.g., [5, 6, 7]) are a "parameter-space" version of weighted-residual (here Galerkin) approximation. To define our reduced-basis procedure, we first introduce a sample set in parameter space, $S^N = \{\mu_1, \ldots, \mu_N\}$, and associated reduced-basis space $W^N = \text{span}\{\zeta_n \equiv u(\mu_n), n = 1, \ldots, N\}$, where $u(\mu_n)$ satisfies (1) for $\mu = \mu_n \in \mathcal{D}$ (note μ^i refers to the i^{th} component of the P–tuple μ, whereas μ_n refers to the n^{th} P–tuple in S^N). We then require our reduced-basis approximation to $u(\mu)$ for any given μ, $u^N(\mu) \in W^N \subset Y$, to satisfy

$$a(u^N(\mu), v; \mu) = \ell(v), \forall v \in W^N; \tag{3}$$

the reduced-basis approximation to $s(\mu)$ can subsequently be evaluated as $s^N(\mu) = \ell^O(u^N(\mu))$.

It is a simple matter to show that

$$\|u(\mu) - u^N(\mu)\|_Y \leq \sqrt{\frac{\gamma}{\alpha}} \min_{w^N \in W^N} \|u(\mu) - w^N\|_Y, \tag{4}$$

which states that our approximation is optimal in the Y norm. It can also be readily shown for our particular problem that

$$s(\mu) = s^N(\mu) + a(e^N(\mu), e^N(\mu); \mu), \tag{5}$$

where $e^N = u - u^N$. It follows from (4),(5), and the continuity of a that

$$|s(\mu) - s^N(\mu)| \leq \frac{\gamma^2}{\alpha} (\min_{w^N \in W^N} \|u(\mu) - w^N\|_Y)^2; \tag{6}$$

thus our output approximation is also optimal.

We must, of course, also understand the extent to which the best w^N in W^N can, indeed, approximate the requisite temperature distribution. The essential point is that, although W^N clearly does not have any approximation properties for *general* functions in Y, simple interpolation arguments in parameter space suggest that W^N should approximate well $u(\mu)$ *even for very modest* N; indeed, exponential convergence is obtained in N for sufficiently smooth μ-dependence (e.g., [6, 7]). It is for this reason that, even in high-dimensional (large P) parameter spaces, reduced-basis methods continue to perform well — indeed, thanks to (6), much better than *ad hoc*, uncontrolled "non-state-space" fits of $(\mu, s(\mu))$ input-output pairs.

We now turn to the computational issues. We first express the reduced-basis approximation as

$$u^N(\boldsymbol{x}; \mu) = \sum_{j=1}^{N} u_j^N(\mu) \zeta_j(\boldsymbol{x}) = (\underline{u}^N(\mu))^T \underline{\zeta}(\boldsymbol{x}), \tag{7}$$

and choose for test functions $v = \zeta_i(\boldsymbol{x}), i = 1, \ldots, N$. We then insert these representations into (3) to yield the desired algebraic equations for $\underline{u}^N(\mu) \in \mathbb{R}^N$,

$$\sum_{j=1}^{N} a(\zeta_j, \zeta_i; \mu) u_j^N = \ell(\zeta_i), \quad i = 1, \ldots, N. \tag{8}$$

Equation (8) can be written in matrix form as

$$\underline{A}(\mu) \underline{u}^N(\mu) = \underline{L}, \tag{9}$$

where $\underline{A}(\mu) \in \mathbb{R}^{N \times N}$ is the SPD matrix with entries $A_{i,j}(\mu) = a(\zeta_j, \zeta_i; \mu), 1 \leq i, j \leq N$, and $\underline{L} \in \mathbb{R}^N$ is the "load" vector with entries $L_i = \ell(\zeta_i), 1 \leq i \leq N$.

We now invoke (2) to note that

$$A_{i,j}(\mu) = a(\zeta_j, \zeta_i; \mu) = \sum_{q=1}^{Q} \sigma^q(\mu) a^q(\zeta_j, \zeta_i) = \sum_{q=1}^{Q} \sigma^q(\mu) A^q_{i,j}, \quad (10)$$

where the matrices $\underline{A}^q \in \mathbb{R}^{N\times N}$ are given by $A^q_{i,j} = a^q(\zeta_j, \zeta_i), 1 \le i, j \le N, q = 1, \ldots, Q$. The off-line/on-line decomposition is now clear. In the *off-line* stage, we construct the $\underline{A}^q, q = 1, \ldots, Q$. In the *on-line* stage, for any given μ, we first form $\underline{A}$ from the $\underline{A}^q$ according to (10); we next invert (9) to find $\underline{u}^N(\mu)$; and we then compute $s^N(\mu) = \ell^O(u^N(\mu)) = \ell(u^N(\mu)) = (\underline{u}^N(\mu))^T \underline{L}$. As we shall see, N will typically be $O(10)$ for our particular problem. Thus, as required, the incremental cost to evaluate $s^N(\mu)$ for any given new μ is very small: $O(N^2Q)$ to form $\underline{A}(\mu)$; $O(N^3)$ to invert (the typically dense) $\underline{A}(\mu)$ system; and $O(N)$ to evaluate $s^N(\mu)$ from $\underline{u}^N(\mu)$.

The above *a priori* results tell us only that we are doing as well as possible; it does not tell us *how* well we are doing. Since the error in our output is not known, the minimal number of basis functions required to satisfy the desired error tolerance can not be ascertained. As a result, either too many or too few functions are retained; the former results in computational inefficiency, the latter in unacceptable uncertainty. We thus need *a posteriori* error bounds as well.

3. Output Bounds

To begin, we assume that we may find a function $g(\mu): \mathcal{D} \to \mathbb{R}_+$ and a symmetric continuous coercive bilinear form $\hat{a}: Y \times Y \to \mathbb{R}$ such that

$$\underline{c}\|v\|^2_Y \le g(\mu)\hat{a}(v, v) \le a(v, v; \mu), \forall v \in Y, \forall \mu \in \mathcal{D}, \quad (11)$$

for some real positive constant $\underline{c}$; for our thermal fin problem we can readily find a $g(\mu)$ and $\hat{a}(w, v)$ such that (11) is satisfied. The procedure is then simple: we first compute $\hat{e}(\mu) \in Y$ solution of

$$g(\mu)\hat{a}(\hat{e}(\mu), v) = R(v; \mu), \forall v \in Y, \quad (12)$$

where $R(v; \mu) \equiv \ell(v) - a(u^N, v; \mu)$ is the residual; we then evaluate our bounds as

$$s^N_-(\mu) = s^N(\mu), \quad s^N_+(\mu) = s^N(\mu) + \Delta^N(\mu), \quad (13)$$

where $\Delta^N(\mu)$, the bound gap, is given by

$$\Delta^N(\mu) = g(\mu)\hat{a}(\hat{e}(\mu), \hat{e}(\mu)). \quad (14)$$

The notion of output bounds is not restricted to reduced-basis approximations: it can also be applied within the context of finite element discretization as well as iterative solution strategies [8, 9].

We can then show that

$$s^N_-(\mu) \le s(\mu) \le s^N_+(\mu), \quad \forall N; \quad (15)$$

we thus have a *certificate of fidelity* for s^N — it is within $\Delta^N(\mu)$ of $s(\mu)$. To prove the left inequality we appeal to (5) and the coercivity of a. To demonstrate the right inequality we first note that $R(e^N(\mu); \mu) = \ell(e^N(\mu)) - a(u^N(\mu), e^N(\mu); \mu) = a(e^N(\mu), e^N(\mu); \mu)$, since $\ell(e^N(\mu)) = a(u, e^N(\mu); \mu)$ from (1) for $v = e^N(\mu)$; we next choose $v = e^N(\mu)$ in (12) to obtain $g(\mu)\hat{a}(\hat{e}(\mu), e^N(\mu)) = a(e^N(\mu), e^N(\mu); \mu)$; then from the right inequality of (11) we have

$$\begin{aligned} \Delta^N(\mu) &\equiv g(\mu)\hat{a}(\hat{e}, \hat{e}) \\ &= g(\mu)\hat{a}(\hat{e} - e^N, \hat{e} - e^N) + 2a(e^N, e^N) \\ &\quad - g(\mu)\hat{a}(e^N, e^N) \\ &\ge g(\mu)\hat{a}(\hat{e} - e^N, \hat{e} - e^N) + a(e^N, e^N); \end{aligned}$$

from the left inequality of (11) we thus conclude that $\Delta^N(\mu) \ge a(e^N, e^N)$; a comparison of (5) and (13) then completes the proof.

We can now ascertain, through Δ^N, the accuracy of our output prediction, which will in turn permit us to adaptively modify our approximation until the prescribed error tolerance ε is satisfied. However, it is also critical that $\Delta^N(\mu)$ be a *good* error estimator; a poor estimator will encourage us to unnecessarily refine an approximation which is, in fact, adequate. To prevent the latter the effectivity $\eta^N(\mu) \equiv \Delta^N(\mu)/|s(\mu) - s^N(\mu)|$ should be order unity. For our problem it is simple to prove that $\eta^N(\mu) \le \gamma/\underline{c}$, independent of μ and N; in practice, effectivities are typically less than 10, which is adequate given the rapid convergence of reduced-basis approximations.

We now turn to the computational issues. From (2) and (7), (12) can be re-written as

$$\hat{a}(\hat{e}(\mu), v) = \frac{1}{g(\mu)}\left(\ell(v) - \sum_{q=1}^{Q}\sum_{j=1}^{N}\sigma^q(\mu)u_j^N(\mu)a^q(\zeta_j, v)\right), \quad \forall v \in Y.$$

We thus see from simple linear superposition that $\hat{e}(\mu)$ can be expressed as

$$\hat{e}(\mu) = \frac{1}{g(\mu)}(\hat{z}_0 + \sum_{q=1}^{Q}\sum_{j=1}^{N}\sigma^q(\mu)u_j^N(\mu)\hat{z}_j^q),$$

where $\hat{z}_0 \in Y$ satisfies $\hat{a}(\hat{z}_0, v) = \ell(v), \forall v \in Y$, and $\hat{z}_j^q \in Y, j = 1, \ldots, N, q = 1, \ldots, Q$, satisfies $\hat{a}(\hat{z}_j^q, v) = -a^q(\zeta_j, v), \forall v \in Y$. It then follows that we can express $\Delta^N(\mu)$ of (14) as

$$\Delta^N(\mu) = \frac{1}{g(\mu)}\Bigg[\underbrace{\hat{a}(\hat{z}_0, \hat{z}_0)}_{c_0} + 2\sum_{q=1}^{Q}\sum_{j=1}^{N}\sigma^q(\mu)u_j^N(\mu)\underbrace{\hat{a}(\hat{z}_0, \hat{z}_j^q)}_{\Lambda_j^q} + \sum_{q=1}^{Q}\sum_{q'=1}^{Q}\sum_{j=1}^{N}\sum_{j'=1}^{N}\sigma^q(\mu)\sigma^{q'}(\mu)u_j^N(\mu)u_{j'}^N(\mu)\underbrace{\hat{a}(\hat{z}_j^q, \hat{z}_{j'}^{q'})}_{\Gamma_{jj'}^{qq'}}\Bigg]; \tag{16}$$

$s_+^N(\mu)$ then directly follows from (13).

The off-line/on-line decomposition is now clear. In the *off-line* stage we compute $\hat{z}_0$ and $\hat{z}_j^q, j = 1, \ldots, N, q = 1, \ldots, Q$, and then the inner products c_0, Λ_j^q, and $\Gamma_{jj'}^{qq'}$ defined in (16). In the *on-line* stage, for any given new μ, and given $s^N(\mu)$ and $\underline{u}^N(\mu)$ as computed in the on-line stage of the output prediction process (Section 2), we evaluate $\Delta^N(\mu)$ as

$$\Delta^N(\mu) = \frac{1}{g(\mu)}\Bigg[c_0 + 2\sum_{q=1}^{Q}\sum_{j=1}^{N}\sigma^q(\mu)u_j^N(\mu)\Lambda_j^q + \sum_{q=1}^{Q}\sum_{q'=1}^{Q}\sum_{j=1}^{N}\sum_{j'=1}^{N}\sigma^q(\mu)\sigma^{q'}(\mu)u_j^N(\mu)u_{j'}^N(\mu)\Gamma_{jj'}^{qq'}\Bigg],$$

and then evaluate $s_+^N(\mu) = s^N(\mu) + \Delta^N(\mu)$. The incremental cost to evaluate $s_+^N(\mu)$ for any given new μ is very small: $O(N^2Q^2)$.

4. Numerical Algorithm

In the simplest case we take our field and output approximations to be $\tilde{u}(\mu) = u^N(\mu)$ and $\tilde{s}(\mu) = s^N(\mu)$, respectively, for some given N, and then compute $\Delta^N(\mu)$ to assess the error. However, we can improve upon this recipe: we take $\tilde{u}(\mu) = u^{\tilde{N}}(\mu)$ and $\tilde{s}(\mu) = s^{\tilde{N}}(\mu)$, where $u^{\tilde{N}}(\mu)$ and $s^{\tilde{N}}(\mu)$ are the reduced-basis approximations associated with a subspace of W^N, $W^{\tilde{N}}$, in which we select only $\tilde{N}$ of our available basis functions. In practice, we include in $W^{\tilde{N}}$ the basis functions corresponding to sample points μ_n closest to the new μ of interest; we continue to (say) double our space until $\Delta^{\tilde{N}}(\mu) \leq \varepsilon$ (and hence $|s(\mu) - s^{\tilde{N}}(\mu)| \leq \varepsilon$). If we satisfy our criterion for $\tilde{N} \leq N$ the adaptive procedure is entirely contained within the on-line stage of the procedure, and the complexity of this stage is reduced from $O(N^2Q + N^3 + N^2Q^2)$ to $O(\tilde{N}^2Q + \tilde{N}^3 + \tilde{N}^2Q^2)$. Note the critical role that our error bound plays in effecting this economy.

In practice — to ensure that the $\zeta_n, \hat{z}_0, \hat{z}_j^q$ are actually calculable — we replace the infinite-dimensional space Y with a very high-dimensional "truth" space Y_T (e.g., a finite element space associated with a very fine triangulation). It follows that we obtain bounds not for s, but rather for $s_T = \ell^O(u_T)$, where $u_T \in Y_T$ satisfies $a(u_T, v; \mu) = \ell(v), \forall v \in Y_T$. The essential point is that Y_T may be chosen very conservatively — and hence the difference between s_T and s rendered arbitrarily small — since (i) the on-line work and storage are in fact *independent* of the dimension of Y_T, $\mathcal{N}$, and (ii) the off-line work will remain modest since N will typically be quite small.

5. Results and Discussion

We first demonstrate the accuracy of the reduced-basis output prediction and output bounds by considering the case $P_{\text{eff}} = 5$ in which $L = 2.5$ and $t = 0.25$ are fixed; the remaining parameters $k^1, k^2, k^3, k^4, \text{Bi}$ vary in $\mathcal{D}_{\text{eff}} \equiv [0.1, 10]^4 \times [0.01, 1]$. The sample points for S^N are chosen randomly (uniformly) over $\mathcal{D}_{\text{eff}}$; the new value of μ to which

we apply the reduced-basis approximation is $k^1 = 0.5, k^2 = 1.0, k^3 = 3.0, k^4 = 9.0, \mathrm{Bi} = 0.6$ (similar results are obtained at other points in $\mathcal{D}_{\text{eff}}$). We present in Table 1 the actual error $|s(\mu) - s^N(\mu)|$; the estimated error $\Delta^N(\mu)$ (our strict upper bound for $|s(\mu) - s^N(\mu)|$); and the effectivity $\eta^N(\mu)$ (the ratio of the estimated and actual errors). We observe the high accuracy and rapid convergence of the reduced-basis prediction, even for this relatively high-dimensional parameter space; and the very good accuracy (low effectivity) of our error bound $\Delta^N(\mu)$. The combination of high accuracy and certifiable fidelity permits us to proceed with an extremely low number of modes.

N	$\lvert s - s^N \rvert$	Δ^N	η^N
10	4.68×10^{-3}	1.43×10^{-2}	3.06
20	4.70×10^{-4}	1.13×10^{-3}	2.40
30	3.04×10^{-4}	1.04×10^{-3}	3.43
40	1.08×10^{-4}	4.61×10^{-4}	4.27
50	2.47×10^{-5}	6.89×10^{-5}	2.78

Table 1

As regards computational cost, in the limit of "infinitely many" evaluations, the calculation of $\tilde{s}(\mu)$ to within 0.1% of s_T is roughly 24 times faster than direct calculation of $s_T = \ell^O(u_T)$; here u_T is our underlying "truth" finite element approximation. The breakeven point at which the reduced-basis approximation first becomes less expensive than direct evaluation of s_T is roughly 250 evaluations. These are fair comparisons: our "truth" approximation here is *not* overly fine, and our solution strategy for $u_T \in Y_T$ (an ILU-preconditioned conjugate-gradient procedure) is quite efficient. The reduced-basis approach is much faster simply because the dimension of W^N, N, is much smaller than the dimension of Y_T, $\mathcal{N}$ (which more than compensates for the loss of sparsity in $\underline{A}$). For more difficult problems that require larger $\mathcal{N}$, or that are not as amenable to fast solution methods on Y_T, the relative efficiency of the reduced-basis approach is even more dramatic.

The obvious advantage of the reduced-basis approach within the design, optimization, and control environment is the very rapid response. However, the "blackbox" nature of the on-line component of the procedure has other advantages. In particular, the on-line code is simple, non-proprietary, and completely decoupled from the (often complicated) off-line "truth" code. This is particularly important in multidisciplinary design optimization, in which various models and approximations must be integrated.

We close this section with a more applied example. We now fix all parameters except L and t, so that $P_{\text{eff}} = 2$; (L, t) vary in $\mathcal{D}_{\text{eff}} = [2.0, 3.0] \times [0.1, 0.5]$. We choose for our two outputs the volume of the fin, $\mathcal{V}$, and the root average temperature, s. As our "design exercise" we now construct the achievable set — all those $(\mathcal{V}, s)$ pairs associated with some (L, t) in $\mathcal{D}_{\text{eff}}$; the result, based on many evaluations of $(\mathcal{V}, s_+^{\tilde{N}})$ for different values of $(L, t) \in \mathcal{D}_{\text{eff}}$, is shown in Figure 2. We present the results in terms of $s_+^{\tilde{N}}$ rather than $s^{\tilde{N}}$ to ensure that the actual temperature s_T will always be lower than our predictions (that is, conservative); and we choose $\tilde{N}$ such that $s_+^{\tilde{N}}$ is always within 0.1% of s_T to ensure that the design process is not misled by inaccurate predictions. Given the obvious preferences of lower volume and lower temperature, the designer will be most interested in the lower left boundary of the achievable set — the Pareto efficient frontier; although this boundary can of course be found without constructing the entire achievable set, many evaluations of the outputs will still be required.

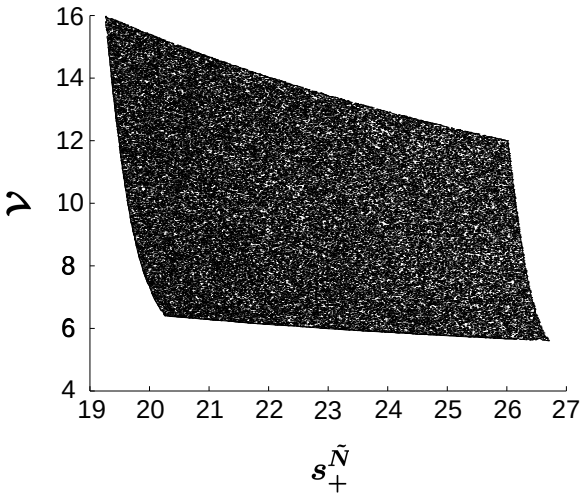

Figure 2

6. Generalizations and Issues

Many of the assumptions that we have introduced are assumptions of convenience and exposition, not necessity. First, the output functional ℓ^O need not be same as the inhomogeneity ℓ; with the introduction of an adjoint (or dual) problem [2], our results

above extend to the more general case. Second, the function $g(\mu)$ need not be known *a priori*: $g(\mu)$ is related to an eigenvalue problem which can itself be readily approximated by a reduced-basis space constructed as the span of appropriate eigenfunctions (in theory we can now only prove asymptotic bounding properties as $N \to \infty$, however in practice the reduced-basis eigenvalue approximation converges very rapidly, and there is thus little loss of certainty). Third, these same notions extend, with some modification, to noncoercive problems, where $g(\mu)$ is now in fact the inf-sup stability parameter [3, 4]. Finally, nonsymmetric operators are readily treated, as are certain classes of nonlinearity in the state variables (e.g., eigenvalue problems [1]).

Perhaps the most limiting assumption is (2), affine dependence on the parameter functions. In some cases (2) may indeed apply, but Q may be rather large. In such cases we can reduce the complexity and storage of the off-line and on-line stages from $O(Q^2)$ to $O(Q)$ by introducing a reduced-basis approximation of the error equation (12) for a suitably chosen "staggered" sample set $S^M_{\rm err}$ and associated reduced-basis space constructed as the span of appropriate error functions. These ideas also extend to the case in which the parameter dependence can not be expressed (or accurately approximated) as in (2); however we now need to at least partially abandon the blackbox nature of the on-line stage of computation, allowing evaluation (though not inversion) of the truth-approximation operator, as well as storage of some reduced-basis vectors of size $\mathcal{N}$. These methods are currently under development. For more information and access to interactive examples, see the web site http://augustine.mit.edu/.

REFERENCES

[1] L. Machiels, Y. Maday, I.B. Oliveira, A.T. Patera, and D.V. Rovas. Output bounds for reduced-basis approximations of symmetric positive definite eigenvalue problems. *C. R. Acad. Sci. Paris, Série I*, to appear.

[2] Y. Maday, L. Machiels, A.T. Patera, and D.V. Rovas. Blackbox reduced-basis output bound methods for shape optimization. In *Proceedings* 12^{th} *International Domain Decomposition Conference*, Chiba Japan, 2000, to appear.

[3] D.V. Rovas. An overview of blackbox reduced-basis output bound methods for elliptic partial differential equations. In *Proceedings* 16^{th} *IMACS World Congress 2000*, Lausanne Switzerland, 2000, to appear.

[4] Y. Maday, A.T. Patera, and D.V. Rovas. A blackbox reduced-basis output bound method for noncoercive linear problems. *MIT-FML Report 00-2-1, 2000; also in the Collège de France Series, to appear.*

[5] A.K. Noor and J.M. Peters. Reduced basis technique for nonlinear analysis of structures. *AIAA Journal*, 18(4):455-462, 1980.

[6] J.P. Fink and W.C. Rheinboldt. On the error behavior of the reduced basis technique in nonlinear finite element approximations. *Z. Angew. Math. Mech.*, 63:21-28, 1983.

[7] T.A. Porsching. Estimation of the error in the reduced basis method solution of nonlinear equations. *Mathematics of Computation*, 45(172):487-496, 1985.

[8] Y. Maday, A.T. Patera, and J. Peraire. A general formulation for a posteriori bounds for output functionals of partial differential equations; application to the eigenvalue problem. *C. R. Acad. Sci. Paris, Série I*, 328:823-829, 1999.

[9] A.T. Patera and E.M. Rønquist. A general output bound result: application to discretization and iteration error estimation and control. *Math. Models Methods Appl. Sci.*, to appear.

The Adjoint Approach in a Nutshell[1]

Robert Michael Lewis
ICASE, Mail Stop 132C NASA Langley Research Center
Hampton, Virginia 23681-2199
buckaroo@icase.edu

1. Introduction

One often hears the term "adjoint approach" in connection with the calculation of first derivatives in the optimization of systems governed by differential equations. We give here a quick introduction to

[1]This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046 while the author was in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681-2199.

the adjoint approach for those unfamiliar with the method.

As an example, consider the differential equation

$$\begin{aligned} -(a(x)u'(x))' - u'(x) &= q(x) \quad x \in (0,1) \\ u(0) = 0, \quad u(1) &= 0. \end{aligned} \tag{1}$$

Given the coefficient $a(x)$, $u(x) = u(a)(x)$ is the solution of (1). Now suppose we wish to find $a(x)$ that minimizes some functional of $u(x)$, say, the least-squares functional

$$F(a) = \frac{1}{2}\int_0^1 (u(a)(x) - u_*(x))^2 dx, \tag{2}$$

where $u_*(x)$ is some given target. Such a functional might arise in parameter estimation, where we are attempting to estimate $a(x)$ from observations $u_*(x)$, or optimal design, where we are attempting to choose $a(x)$ in order to obtain some preferred behavior $u_*(x)$. The question we consider is how one computes the derivative $F'(a)$ of $F(a)$ with respect to a for use in a derivative-based optimization technique.

Abstractly, we have a nonlinear program of the form

$$\begin{aligned} \underset{a}{\text{minimize}} \quad & F(a) = f(a, u(a)) \\ \text{subject to} \quad & C(a) = c(a, u(a)) \geq 0, \end{aligned} \tag{3}$$

where, given the *design variables* a, we compute the *state variable* $u(a)$ by solving some manner of differential equation

$$S(a, u(a)) = 0. \tag{4}$$

We can take advantage of the structure introduced by (4) when computing derivatives of F and C.

2. A quick and dirty calculation

We proceed formally. By the chain rule,

$$F'(a) = f_a + f_u \frac{du}{da} \quad , \quad C'(a) = c_a + c_u \frac{du}{da} \quad .$$

Implicit differentiation of (4), yields

$$\frac{du}{da} = -S_u^{-1} S_a.$$[2]

Thus,

$$\begin{aligned} F'(a) &= f_a - f_u S_u^{-1} S_a \\ C'(a) &= c_a - c_u S_u^{-1} S_a. \end{aligned} \tag{5}$$

3. The sensitivity equations

Formula (5) corresponds to what is known as the *sensitivity equations* approach to computing derivatives. In a computational setting, suppose that there are

$$\begin{aligned} &n \text{ design variables:} && a = (a_1, \cdots, a_n), \\ &m \text{ state variables:} && u = (u_1, \cdots, u_m), \\ &p \text{ constraints:} && c = (c_1, \cdots, c_p). \end{aligned}$$

Then the dimensions of the matrices in the product (5) are

$$\begin{aligned} \underbrace{F'(a)}_{1\times n} &= \underbrace{f_a}_{1\times n} - \underbrace{f_u}_{1\times m}\underbrace{S_u^{-1}}_{m\times m}\underbrace{S_a}_{m\times n} \\ \underbrace{C'(a)}_{p\times n} &= \underbrace{c_a}_{p\times n} - \underbrace{c_u}_{p\times m}\underbrace{S_u^{-1}}_{m\times m}\underbrace{S_a}_{m\times n}. \end{aligned} \tag{6}$$

This calculation of $F'(a)$ and $C'(a)$ entails multiplication of the n columns of S_a by S_u^{-1}. That is, we must solve the linearized state equation (represented by the linearized operator S_u) for n different right-hand sides. Thus, the cost of computing $F'(a)$ via (6) grows with the number of *inputs* (independent variables) a. For those readers familiar with automatic differentiation, we note that this order of calculations corresponds to the forward mode of automatic differentiation.

[2] The appeal to the implicit function theorem requires justification in the general setting of the dependence on a of $u(a)$, since a may represent such quantities as coefficients in differential operators, boundary values, or the geometry of the domain on which the problem is posed. In practice, it is sometimes difficult and even impossible to verify that the objective and constraints are truly differentiable.

4. The adjoint approach

For a finite-dimensional problem, we can transpose (6) to obtain

$$\begin{aligned} \underbrace{\nabla F(a)}_{n\times 1} &= \underbrace{\nabla_a f}_{n\times 1} - \underbrace{S_a^T}_{n\times m}\underbrace{S_u^{-T}}_{m\times m}\underbrace{\nabla_u f}_{m\times 1} \qquad (7)\\ \underbrace{\nabla C(a)}_{n\times p} &= \underbrace{\nabla_a c}_{n\times p} - \underbrace{S_a^T}_{n\times m}\underbrace{S_u^{-T}}_{m\times m}\underbrace{\nabla_u c}_{m\times p} \end{aligned}$$

This calculation requires only *one* application of S_u^{-T} to compute $F'(a)$, and p applications of S_u^{-T} to compute $C'(a)$. After transposition of (6), the cost thus depends on the number of *outputs*. This order of calculations corresponds to the reverse mode of automatic differentiation.

The adjoint approach is the infinite-dimensional analog of this transposition. The infinite-dimensional situation is more complicated, since $F'(a)$, f_a, f_u, etc., are now linear functionals on possibly infinite-dimensional spaces of functions. Nevertheless, the transposition in (7) is the basic idea underlying the adjoint approach.

The quantity $\lambda = -S_u^{-T}\nabla_u f$ that arises in computing $\nabla_a F(a)$ is called the *costate*. One similarly computes costates for the columns of $\nabla_a C(a)$.

5. The model problem

We illustrate the adjoint approach for the model problem (1)–(2). It is convenient to view (1) in the weak form, so that $u \in H_0^1[0,1]$ and

$$\begin{aligned} &\int_0^1 [a(x)u'(x)v'(x) + u(x)v'(x)]\ dx \qquad (8)\\ &\quad - \int_0^1 q(x)v(x)\ dx = 0 \end{aligned}$$

for all $v \in H_0^1$. Then $S(a, u(a)) = 0$ is defined by (8). Note that S takes its value in the space of linear functionals acting on $v \in H_0^1$.

The adjoint equation $S_u^T\lambda = -f_u$ is then given by [3]

$$\begin{aligned} &\langle S_u^T\lambda,\ v\rangle \equiv \langle \lambda,\ S_u v\rangle \qquad (9)\\ &\quad = \int_0^1 [a(x)u'(x)\lambda'(x) + u(x)\lambda'(x)]\ dx\\ &\quad = \langle -f_u,\ v\rangle = -\int_0^1 (u(x) - u_*(x))v(x)\ dx \end{aligned}$$

for all $v \in H_0^1$, where $\langle\cdot,\ \cdot\rangle$ represents the duality pairing between linear functionals and the arguments they act on. We recognize (9) as the weak form of the *adjoint problem*

$$\begin{aligned} -(a(x)\lambda'(x))' + \lambda'(x) &= -(u(x) - u_*(x))\\ \lambda(0) = \lambda(1) &= 0. \end{aligned}$$

Finally, $F'(a) = S_a^T\lambda$. We have

$$\begin{aligned} F'(a)\eta &= \langle S_a^T\lambda,\ \eta\rangle \equiv \langle \lambda,\ S_a\eta\rangle\\ &= \int_0^1 \eta(x)u'(x)\lambda'(x)\ dx. \end{aligned}$$

This gives a formula and a representation for the linear functional $F'(a)$ in terms of integration against the function $\lambda'(x)u'(x)$. However, the function $\lambda'(x)u'(x)$ may or may not be suitable as a direction of steepest descent, and may require further smoothing depending on the application.

6. Some comments

Note that the adjoint approach is more efficient when one has fewer dependent variables (i.e., outputs F and C) than independent variables (design variables a), while the sensitivity equations are more efficient if the situation is reversed. If one has a large number of both inputs and outputs, then one approach or the other may be more efficient relative to the other, but either will be expensive in absolute terms.

What about second derivatives? Unfortunately, there is no especially efficient way to compute second derivatives. The analytical calculation of a second derivative requires an application of both S_u^{-1} and S_u^{-T}.

Finally, we allude to interesting problems that can arise in connection with the costate λ and the adjoint problem. The costate λ is actually a linear functional, and as such, it can "look" rather strange if one tries to interpret it as a function in the usual sense (for instance, consider the delta function). For

[3] For a linear map $L : X \to Y$ between two linear spaces X and Y, the adjoint L^T is a map $L^T : X \to Y$ between the dual spaces Y' and X' defined by $\langle L^T\lambda,\ v\rangle = \langle \lambda,\ Lv\rangle$ for all $\lambda \in Y'$ and $v \in X$. The appearance of the dual spaces of infinite-dimensional spaces is what sometimes makes the interpretation of the transposition in (7) a bit subtle.

some problems, the nature of λ can interact with the definition of the adjoint problem to make some approaches to deriving the adjoint problem break down. This in turn, has led some authors to conclude that the adjoint approach cannot be applied to their problem. This is, in fact, not the case, if the nature of λ and the adjoint problem are properly understood. For a discussion of these subtleties, see [4].

7. Historical note

To the author's knowledge, the earliest uses in print of the technique we now call the adjoint approach appear in [3, 2]. The adjoint approach is sometimes attributed to Lions in [5], but what Lions uses is the Karush-Kuhn-Tucker system of necessary conditions for (3)–(4) (absent constraints) formulated as

$$\begin{array}{ll} \underset{a,u}{\text{minimize}} & f(a,u) \\ \text{subject to} & S(a,u) = 0. \end{array} \tag{10}$$

Treatment of (4) as equality constraints in the context of generalized reduced gradient algorithms also led early on (e.g., [1]) to techniques that may be viewed as antecedents of the adjoint approach. Indeed, the reader familiar with variable reduction techniques will note that the formulae (6) and (7) correspond to formulae for the reduced gradient of f with respect to the equality constraints $S(a,u) = 0$ in (10).

REFERENCES

[1] J. Abadie, *Application of the GRG algorithm to optimal control problems*, in Integer and Nonlinear Programming, J. Abadie, ed., North-Holland Elsevier, 1970.

[2] G. Chavent, M. Dupuy, and P. Lemonnier, *History matching by use of optimal theory*, Society of Petroleum Engineers Journal, (1975), pp. 74–86.

[3] W. H. Chen, G. R. Gavalas, J. H. Seinfeld, and M. L. Wasserman, *A new algorithm for automatic history matching*, Society of Petroleum Engineers Journal, (1974), pp. 593–608.

[4] R. M. Lewis, *Numerical computation of sensitivities and the adjoint approach*, in Computational Methods for Optimal Design and Control, J. Borggaard, J. Burns, E. Cliff, and S. Schreck, eds., Birkhäuser, 1998, pp. 285–302. Also available as ICASE technical report 97–61.

[5] J. L. Lions, *Contrôle optimal de systèmes gouvernés par des équations aux dérivées partielles*, Dunod, 1968.

A Lagrange-Newton-Krylov-Schur Method for PDE-Constrained Optimization

George Biros and Omar Ghattas
Mechanics, Algorithms, and Computing Laboratory
Department of Civil & Environmental Engineering
Carnegie Mellon University, Pittsburgh, PA, USA
Email: {biros,oghattas}@cs.cmu.edu
URL: http://www.cs.cmu.edu/~{gbiros,oghattas}

1. Introduction

PDE-constrained optimization is a frontier problem in computational science and engineering. All PDE-constrained problems share the difficulty that PDE solution is just a subproblem associated with optimization. Thus, the optimization problem is often significantly more difficult to solve than the simulation problem. We are particularly interested in large-scale problems that require parallel computing to make them tractable.

To illustrate the main issues, let's consider a model problem of optimal distributed control of a Navier-Stokes flow:

$$\min \mathcal{F}(\boldsymbol{u}, p, \boldsymbol{b}) =$$
$$\frac{1}{2}\int_\Omega (\boldsymbol{\nabla u} + \boldsymbol{\nabla u}^T) : (\boldsymbol{\nabla u} + \boldsymbol{\nabla u}^T) d\Omega + \frac{\alpha}{2}\int_\Omega \boldsymbol{b}\cdot\boldsymbol{b}\, d\Omega$$

subject to:

$$\begin{aligned} -\nu\boldsymbol{\nabla}\cdot(\boldsymbol{\nabla u} + \boldsymbol{\nabla u}^T) + (\boldsymbol{\nabla u})\boldsymbol{u} + \boldsymbol{\nabla} p + \boldsymbol{b} &= \boldsymbol{0} \quad \text{in} \quad \Omega \\ \boldsymbol{\nabla}\cdot\boldsymbol{u} &= 0 \quad \text{in} \quad \Omega \\ \boldsymbol{u} &= \boldsymbol{0} \quad \text{on} \quad \Gamma \end{aligned}$$

Here, $\boldsymbol{u}$ is the fluid velocity field, p the pressure field, $\boldsymbol{b}$ the body force control function, α a weighting parameter, and ν the inverse of the Reynolds number. The objective is to minimize the rate of dissipation of viscous energy and a cost associated with a body force control function. The constraints are the stationary incompressible Navier-Stokes equations with Dirichlet boundary conditions.

We can form a Lagrangian functional, and require its stationarity with respect to the state ($\boldsymbol{u}$,p) and optimization ($\boldsymbol{b}$) variables and the Lagrange multipliers. Taking variations and invoking the appropriate Green identities, we arrive at the following first-order necessary conditions:

Adjoint Equations:

$$\begin{aligned} -\nu\boldsymbol{\nabla}\cdot(\boldsymbol{\nabla}\boldsymbol{\lambda}_u+\boldsymbol{\nabla}\boldsymbol{\lambda}_u^T)+(\boldsymbol{\nabla}\boldsymbol{u})^T\boldsymbol{\lambda}_u-(\boldsymbol{\nabla}\boldsymbol{\lambda}_u)\boldsymbol{u} \\ +\boldsymbol{\nabla}\lambda_p-\boldsymbol{\nabla}\cdot(\boldsymbol{\nabla}\boldsymbol{u}+\boldsymbol{\nabla}\boldsymbol{u}^T)=\boldsymbol{0} \quad \text{in} \quad \Omega \\ \boldsymbol{\nabla}\cdot\boldsymbol{\lambda}_u=\boldsymbol{0} \quad \text{in} \quad \Omega \\ \boldsymbol{\lambda}_u=\boldsymbol{0} \quad \text{on} \quad \Gamma \end{aligned}$$

State Equations:

$$\begin{aligned} -\nu\boldsymbol{\nabla}\cdot(\boldsymbol{\nabla}\boldsymbol{u}+\boldsymbol{\nabla}\boldsymbol{u}^T)+(\boldsymbol{\nabla}\boldsymbol{u})\boldsymbol{u}+\boldsymbol{\nabla}p+\boldsymbol{b}=\boldsymbol{0} \quad \text{in} \quad \Omega \\ \boldsymbol{\nabla}\cdot\boldsymbol{u}=\boldsymbol{0} \quad \text{in} \quad \Omega \\ \boldsymbol{u}=\boldsymbol{0} \quad \text{on} \quad \Gamma \end{aligned}$$

Control Equations:

$$\rho\,\boldsymbol{b}+\boldsymbol{\lambda}_u=\boldsymbol{0} \quad \text{in} \quad \Omega$$

The state equations are just the original Navier-Stokes PDEs. The *adjoint equations*, which result from stationarity with respect to state variables, are themselves PDEs, and are linear in the Lagrange multipliers $\boldsymbol{\lambda}_u$ and λ_p. Finally, the *control equations* are (in this case) algebraic.

Thus we end up with a large, coupled, unstructured system of optimality conditions (or at least bigger, more coupled, and less structured than seen by a Navier-Stokes solvers). How to go about solving it? The usual way is to eliminate state variables and Lagrange multipliers and, correspondingly, the state equations and adjoint equations; to reduce the system to a manageable one in just the control (i.e. decision) variables $\boldsymbol{b}$. Here's one way to do this: given $\boldsymbol{b}$ at some iteration, we solve the state equations for the state variables $\boldsymbol{u}, p$. Knowing the state variables then permits us to solve the adjoint equations for the Lagrange multipliers $\boldsymbol{\lambda}_u, \lambda_p$. Finally, with the states and multipliers known, we can update $\boldsymbol{b}$ by iterating on the control equation. The whole process is repeated until convergence. This elimination procedure is termed a *reduced space* method, in contrast to a *full space* method, in which one solves for the states, controls, and multipliers simultaneously.

Reduced space methods are attractive for several reasons. Solving the subsets of equations in sequence imparts some structure to the problem. State equation solvers build on years of development of large-scale parallel PDE solvers. Adjoint PDE solvers don't exactly grow on trees—but the strong similarities between the state and adjoint operators suggest that an existing PDE solver for the state equations can be modified easily to handle the adjoint system (at least on a good day). Finally, the control equations are usually reasonably tame, at least to evaluate. Another advantage of reduction is that the full space system is often very ill-conditioned, whereas the three subsystems are typically better conditioned.

On the other hand, the big disadvantage of reduced methods is the need to solve the state and adjoint equations *at each iteration* of the reduced system—a direct consequence of the reduction onto the decision variable space. So it's natural to go back to the full space, and ask if it's possible to solve the entire optimality system simultaneously, but retain the structure-inducing, condition-improving advantages of reduced space methods—while avoiding their disadvantages.

In this article, we present such a method. The key idea is to solve in the full space using a Newton method, but precondition with a quasi-Newton reduced space method. The Karush-Kuhn-Tucker system arising at each Newton iteration is solved using a Krylov iterative method, and it is this system to which the preconditioner is applied. We have found that the reduced space preconditioner is very effective in reducing the number of Krylov iterations, and applying it captures the favorable structure of reduced methods. On the other hand, since the reduction is used just as a preconditioner, we can cheat on the state and adjoint solves, replacing them with approximations which could be their own preconditioners. So we arrive at a method that combines rapid convergence in the outer Newton iteration (typically mesh-independent), with fast convergence of the inner Krylov iteration (which can be as good as mesh-independent). We don't even need to compute second derivatives—since a Krylov method is used to solve the KKT system, we can apply the usual directional differencing trick to approximate the Lagrangian Hessian–vector product.

Why the name *Lagrange-Newton-Krylov-Schur*? It is common in PDE-solver circles to use the phrase *Newton-Krylov-X* to refer to Newton methods for solving PDEs that employ Krylov linear solvers, with X as the preconditioner for the Krylov method. Since *Lagrange-Newton* is sometimes used to describe a Newton method for solving the optimality system (a.k.a. an SQP method), and since a reduced space method can be viewed as a Schur complement method for the KKT system, we arrive at the concatenation *LNKS*. It's a mouthful, but it preserves the tie to modern PDE solvers, whose use of approximate decompositions as preconditioners inspired this approach [6]. David Keyes suggested (a variation of) this name in his plenary talk at the 1999 combined SIAM Optimization/Annual meeting [5].

In the remainder of this article, we give a brief overview of the LNKS method and some sample results for an optimal flow control problem on a Cray T3E. Further details can be found in [2], and more extensive discussion and results in forthcoming articles that focus on the inner Krylov iteration [3] and the outer Newton iteration [4]. We note finally that Battermann and Heinkenschloss have presented a somewhat different method for preconditioning KKT matrices that also makes use of state and control space decompositions [1].

2. Reduced Space Methods

In this section we discuss reduced space SQP methods, concentrating on the discrete form of a typical PDE-constrained optimization problem:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0},$$

where $\boldsymbol{x}$ are the state and decision variables, f is the objective function and $\boldsymbol{c}$ are the discretized state equations. Using Lagrange multipliers $\boldsymbol{\lambda}$, we can define the Lagrangian function by

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) := f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{c}(\boldsymbol{x}).$$

The first order optimality conditions require that the Lagrangian gradient vanish:

$$\left\{ \begin{array}{c} \boldsymbol{\partial}_x \mathcal{L} \\ \boldsymbol{\partial}_\lambda \mathcal{L} \end{array} \right\} = \left\{ \begin{array}{c} \boldsymbol{g} + \boldsymbol{A}^T \boldsymbol{\lambda} \\ \boldsymbol{c} \end{array} \right\} = \boldsymbol{0},$$

where $\boldsymbol{g}$ is the gradient of the f and $\boldsymbol{A}$ is the Jacobian matrix of the constraints. A Newton step on the optimality conditions (which, in the absence of inequality constraints, is Sequential Quadratic Programming) is given by:

$$\left[\begin{array}{cc} \boldsymbol{W} & \boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{0} \end{array} \right] \left\{ \begin{array}{c} \boldsymbol{p}_x \\ \boldsymbol{\lambda}_+ \end{array} \right\} = - \left\{ \begin{array}{c} \boldsymbol{g} \\ \boldsymbol{c} \end{array} \right\},$$

where $\boldsymbol{W}$ is the Hessian of the Lagrangian function with respect to the optimization variables, $\boldsymbol{p}_x$ is the search direction in $\boldsymbol{x}$, and $\boldsymbol{\lambda}_+$ is the updated Lagrange multiplier. This system is known as the Karush-Kuhn-Tucker (KKT) system, and its coefficient matrix as the KKT matrix. To exploit the structure of the state constraints, we partition the optimization variables into state variables $\boldsymbol{x}_s$ and decision variables $\boldsymbol{x}_d$. The partitioned KKT system becomes:

$$\left[\begin{array}{ccc} \boldsymbol{W}_{ss} & \boldsymbol{W}_{sd} & \boldsymbol{A}_s^T \\ \boldsymbol{W}_{ds} & \boldsymbol{W}_{dd} & \boldsymbol{A}_d^T \\ \boldsymbol{A}_s & \boldsymbol{A}_d & \boldsymbol{0} \end{array} \right] \left\{ \begin{array}{c} \boldsymbol{p}_s \\ \boldsymbol{p}_d \\ \boldsymbol{\lambda}_+ \end{array} \right\} = - \left\{ \begin{array}{c} \boldsymbol{g}_s \\ \boldsymbol{g}_d \\ \boldsymbol{c} \end{array} \right\} \quad (1)$$

This system is of dimension $2n + m$, where n is the number of state variables and m the number of decision variables. State-of-the-art algorithms for PDE-constrained optimization exploit two facts. First, nobody wants to compute second derivatives—it's hard enough convincing the PDE solver community of the need for first derivatives. (No doubt this difficulty will be mitigated by continuing advances in automatic differentiation tools.) And second, everybody wants to use existing software for "inverting" the state Jacobian. Since this is the kernel step in a Newton-based PDE solver, there is a large body of work to draw from. For example, for elliptic PDEs, there exist optimal or nearly-optimal parallel algorithms (e.g. domain decomposition methods or multigrid) that require algorithmic work that is linear or weakly superlinear in n, and scale to thousands of processors and millions of variables.

One way to exploit existing PDE-solvers is to eliminate the state and adjoint equations and variables, and then solve an unconstrained optimization problem in the remaining decision space (this is similar to the argument of the previous section, except here we are linearizing first, then eliminating, as opposed to vice versa.) We refer to this as *Newton reduced SQP* (or N-RSQP), and it can be derived

by block elimination on the KKT system: Given $\boldsymbol{p}_d$, solve the last block of equations (the state system) for $\boldsymbol{p}_s$; then solve the first (the adjoint system) to find $\boldsymbol{\lambda}_+$, and finally solve the middle (the decision system) for $\boldsymbol{p}_d$. It is easy to verify that this block elimination is equivalent to the following block factorization of the KKT matrix:

$$\begin{bmatrix} \boldsymbol{W}_{ss}\boldsymbol{A}_s^{-1} & \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{W}_{ds}\boldsymbol{A}_s^{-1} & \boldsymbol{I} & \boldsymbol{A}_d^T\boldsymbol{A}_s^{-T} \\ \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_s & \boldsymbol{A}_d & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W}_z & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{W}_{yz} & \boldsymbol{A}_s^T \end{bmatrix} \tag{2}$$

where the reduced Hessian matrix is defined by

$$\begin{aligned} \boldsymbol{W}_z := {} & \boldsymbol{A}_d^T\boldsymbol{A}_s^{-T}\boldsymbol{W}_{ss}\boldsymbol{A}_s^{-1}\boldsymbol{A}_d \\ & -\boldsymbol{A}_d^T\boldsymbol{A}_s^{-T}\boldsymbol{W}_{sd} - -\boldsymbol{W}_{ds}\boldsymbol{A}_s^{-1}\boldsymbol{A}_d + \boldsymbol{W}_{dd}, \end{aligned}$$

and the "cross-Hessian" by

$$\boldsymbol{W}_{yz} := \boldsymbol{W}_{sd} - \boldsymbol{W}_{ss}\boldsymbol{A}_s^{-1}\boldsymbol{A}_d.$$

Note that these factors can be permuted to block triangular form, so we can think of this as a block LU factorization of the KKT matrix. It is clear that the only linear systems that need to be solved have either the state Jacobian $\boldsymbol{A}_s$ or its transpose as their coefficient matrix—a "solved problem"—or else the reduced Hessian $\boldsymbol{W}_z$, which is dense and of dimension of the decision space. Thus, reduced methods are particularly attractive when the the decision variables are much fewer than the states.

But two problems remain. First are the second derivative terms. Second, and more problematic, is the need for m solutions of the (linearized) state equations for construction of $\boldsymbol{A}_s^{-1}\boldsymbol{A}_d$ in $\boldsymbol{W}_z$. This is particularly troublesome for large-scale 3D problems, where (linearized) PDE systems are usually solved iteratively, and solution costs cannot be amortized over multiple right hands as effectively as with direct solvers. When the simulation problem is an overnight run on a large parallel machine, this requirement effectively rules out the use of N-RSQP.

A popular technique that addresses these two difficulties is a quasi-Newton RSQP (QN-RSQP) method that replaces the reduced Hessian $\boldsymbol{W}_z$ with a quasi-Newton approximation $\boldsymbol{B}_z$, and discards all other Hessian terms. This corresponds to the following approximation of the KKT block factors:

$$\begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{A}_d^T\boldsymbol{A}_s^{-T} \\ \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_s & \boldsymbol{A}_d & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_z & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{A}_s^T \end{bmatrix} \tag{3}$$

It is easy to verify that just two state solves per iteration are required (actually one linearized state, and one adjoint), as opposed to the m of N-RSQP. And with Hessian terms either approximated or dropped, no second derivatives are needed. A measure of the success of QN-RSQP is its application to numerous optimal control, optimal design, and inverse problems governed by PDEs from linear and nonlinear elasticity, incompressible and compressible flow, heat conduction and convection, phase changes, flow through porous media, etc. Of course, something has to give, and that is the convergence rate: a reduction from quadratic in the Newton case to two-step superlinear. Moreover, the number of iterations taken by QN-RSQP depends on the conditioning of the reduced Hessian, and often increases as the number of decision variables grows, rendering large-scale problems intractable. In the next section, we propose a method that combines the fast convergence of Newton's method with the structure-exploiting properties of reduced methods.

3. LNKS: Krylov solution of the KKT system with approximate QN-RSQP preconditioning

In this section, we present a method for solving the KKT system (1). For optimization problems constrained by 3D PDEs, sparse factorization of the KKT matrix is not an option—such methods are not viable for $\boldsymbol{A}_s$, let alone the entire matrix. Instead, we use a Krylov iterative method, specifically the quasi-minimum residual (QMR) method. However, the varying scales between Hessian and Jacobian terms in the KKT matrix, and its indefiniteness, demand an effective preconditioner. This preconditioner must be capable of exploiting the structure of the state constraints (specifically that good preconditioners exist for $\boldsymbol{A}_s$), must be cheap to apply, and must be effective in reducing the number of Krylov iterations. The QN-RSQP method described in the

previous section fits the bill. Applying the preconditioner amounts to solving with the QN-RSQP factorization (3), except that state Jacobians are replaced by their approximations $\tilde{\boldsymbol{A}}_s$:

$$\begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} \\ \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{A}_d^T \tilde{\boldsymbol{A}}_s^{-T} \\ \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{A}}_s & \boldsymbol{A}_d & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}_z & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \tilde{\boldsymbol{A}}_s^T \end{bmatrix} \tag{4}$$

Replacing $\boldsymbol{A}_s$ with $\tilde{\boldsymbol{A}}_s$ is permissible, since QN-RSQP is being used as a preconditioner. A good choice for $\tilde{\boldsymbol{A}}_s$ is, in turn, one of the available preconditioners for $\boldsymbol{A}_s$—for many PDE operators, there exist near-spectrally-equivalent preconditioners that are both cheap to apply (typically linear or weakly superlinear in problem size) and effective (resulting in iteration numbers that are independent of, or increase very slowly in, problem size).

With (4) used as a preconditioner, the preconditioned KKT matrix ends up having the following form:

$$\begin{bmatrix} \boldsymbol{I}_s & \mathcal{O}(\boldsymbol{E}_s) & \boldsymbol{0} \\ \tilde{\boldsymbol{W}}_{yz}^T \tilde{\boldsymbol{A}}_s^{-1} & \mathcal{O}(\boldsymbol{E}_s) + \boldsymbol{W}_z \boldsymbol{B}_z^{-1} & \mathcal{O}(\boldsymbol{E}_s) \\ \boldsymbol{W}_{ss} \tilde{\boldsymbol{A}}_s^{-1} & \tilde{\boldsymbol{W}}_{yz} \boldsymbol{B}_z^{-1} & \boldsymbol{I}_s \end{bmatrix}$$

where

$$\begin{aligned} \boldsymbol{E}_s &:= \boldsymbol{A}_s^{-1} - \tilde{\boldsymbol{A}}_s^{-1} \\ \boldsymbol{I}_s &:= \boldsymbol{A}_s \tilde{\boldsymbol{A}}_s^{-1} \\ \tilde{\boldsymbol{W}}_{yz} &:= \boldsymbol{W}_{sd} - \boldsymbol{W}_{ss} \tilde{\boldsymbol{A}}_s^{-1} \boldsymbol{A}_d \end{aligned}$$

For exact state equation solution, $\boldsymbol{E}_s = \boldsymbol{0}$ and $\boldsymbol{I}_s = \boldsymbol{I}$, and we see that the QN-RSQP preconditioner clusters the spectrum of the KKT matrix, with all eigenvalues either unit or belonging to $\boldsymbol{W}_z \boldsymbol{B}_z^{-1}$. Therefore, when $\tilde{\boldsymbol{A}}_s$ is a good preconditioner for the state Jacobian, and when $\boldsymbol{B}_z$ is a good approximation of the reduced Hessian (as it should be asymptotically), we might expect the QN-RSQP preconditioner (4) to be effective in reducing the number of Krylov iterations (but note that the preconditioned KKT matrix is non-normal, so a rigorous analysis requires well-conditioned eigenvectors).

How scalable is the method, with respect to increasing problem size and number of processors? For scalability, we require that the work increase near-linearly with problem size (algorithmic scalability) and that it parallelizes well (parallel scalability). Let us examine the major components:

Formation of the KKT matrix–vector product. For PDE-constrained optimization, the Hessian of the Lagrangian function and the Jacobian of the constraints are usually sparse with structure dictated by the mesh (particularly when the decision variables are mesh-related). Thus, formation of the matrix-vector product at each QMR iteration is linear in both state and decision variables, and parallelizes well due to a high computation-to-communication ratio and minimal sequential bottlenecks.

Application of the QN-RSQP preconditioner. The main work involved is application of the state Jacobian preconditioner $\tilde{\boldsymbol{A}}_s$ and its transpose, and "inversion" of the quasi-Newton approximation to the reduced Hessian, $\boldsymbol{B}_z$. We can often make use of scalable, parallel state Jacobian preconditioners that requires $\mathcal{O}(n)$ work to apply (as in various domain decomposition preconditioners for elliptic problems). Furthermore, when $\boldsymbol{B}_z$ is based on a limited-memory quasi-Newton update (as in our implementation), its work is also linear in the decision variables, and the vector operations are easily parallelized (or as easily as vector inner products can be). Therefore, we conclude that application of the QN-RSQP preconditioner requires linear work and parallelizes well.

The Krylov (inner) iteration. As argued above, with an "optimal" state preconditioner and a good $\boldsymbol{B}_z$ approximation, we can anticipate that the number of inner, Krylov iterations will be relatively insensitive to the problem size.

The Lagrange-Newton (outer) iteration. The number of outer, Newton iterations is often independent of problem size for PDE-type problems, and the problems we have solved exhibit this type of behavior as well.

This combination of linear work per Krylov iteration, weak dependence of Krylov iterations on problem size, and independence of Lagrange-Newton iterations on problem size suggest a method that scales well with increasing problem size and number of processors.

How well does the LNKS method work in practice? Here, we quote a set of representative results from many we have obtained for up to 1.5 million state variables and 50,000 control variables on up

to 256 processors. The problem is optimal Navier-Stokes flow control, similar to that of Section 1, except that the controls are boundary velocities. The specific problem is control of 3D flow around a cylinder at subcritical conditions, with controls on the downstream side of the cylinder. Approximation is by Galerkin finite elements, both for state and control variables. We have implemented the LNKS method on top of the PETSc library for parallel solution of PDEs from Argonne. The table shows results for 64 and 128 processors of a Cray T3E for a roughly doubling of problem size. Results for the QN-RSQP and LNKS algorithms are presented. In the table LNKS-EX refers to exact solution of the linearized Navier-Stokes equation within the QN-RSQP preconditioner, whereas LNKS-PR refers to application of a block-Jacobi (with local ILU(0)) approximation of the linearized Navier-Stokes operator. LNKS-PR-TR uses a truncated Newton method and avoids fully converging the KKT system for iterates that are far from a solution.

The results in the table reflect the independence of Newton iterations on problem size, the mild dependence of KKT iterations on problem size, and the resulting reasonable scalability of the method. It is important to point out here that the Navier-Stokes discrete operator is very ill-conditioned, and there is room for improvement of its domain-decomposition preconditioner. The performance of the QN-RSQP KKT preconditioner would improve correspondingly. A dramatic acceleration of the LNKS algorithm is achieved by truncating the Krylov iterations.

states controls	**preconditioning**	**Newton iter**	**average KKT iter**	**time (hours)**
389,440	QN-RSQP	189	—	46.3
6,549	LNKS-EX	6	19	27.4
(64 procs)	LNKS-PR	6	2,153	15.7
	LNKS-PR-TR	13	238	3.8
615,981	QN-RSQP	204	—	53.1
8,901	LNKS-EX	7	20	33.8
(128 procs)	LNKS-PR	6	3,583	16.8
	LNKS-PR-TR	12	379	4.1

More detailed results are given in [2, 3, 4]. These references also discuss the important topics of globalization and the details of the inexactness in solving the KKT system, which were not mentioned here for reasons of space. Another issue is additional inequality constraints; we have recently implemented with Andreas Wächter and Larry Biegler a parallel version of their interior point method for treating such constraints, within the context of LNKS. Finally, this summer we will be releasing a publicly-available software library for parallel solution of PDE-constrained optimization problems, built on top of the PETSc system, and including LNKS and other methods.

Acknowledgments

This work is a part of the Terascale Algorithms for Optimization of Simulations (TAOS) project at CMU, with support from NASA grant NAG-1-2090 and NSF grant ECS-9732301 (under the NSF/Sandia Life Cycle Engineering Program). Computing services on the Pittsburgh Supercomputing Center's Cray T3E were provided under PSC grant ASC-990003P. We thank Satish Balay, Bill Gropp, Lois McInnes, and Barry Smith of Argonne National Lab for their work in making PETSc available to the research community. We also thank Jonathan Shewchuk of UC Berkeley for providing the meshing and partitioning routines Pyramid and Slice. Finally, we thank David Keyes of Old Dominion University/ICASE/Lawrence Livermore, David Young of Boeing, and other members of the TAOS project—Roscoe Bartlett, Larry Biegler, Ivan Malčević, Andreas Wächter—for their useful comments.

REFERENCES

[1] A. Battermann and M. Heinkenschloss, *Preconditioners for Karush–Kuhn–Tucker matrices arising in optimal control of distributed systems*, Tech. Rep. TR96-34, Department of Computational and Applied Mathematics, Rice University, November 1996.

[2] G. Biros and O. Ghattas, *Parallel Newton-Krylov algorithms for PDE–constrained optimization*, in Proceedings of SC99, Portland, Oregon, 1999. Available from http://cs.cmu.edu/~oghattas/papers/sc99/sc99.ps.

[3] ——, *Lagrange-Newton-Krylov-Schur algorithms for PDE-constrained optimization. Part I: KKT preconditioners*, 2000. In preparation.

[4] ——, *Lagrange-Newton-Krylov-Schur algorithms for PDE-constrained optimization. Part II: Truncated Newton solver*, 2000. In preparation.

[5] D. E. Keyes, *Krylov, Lagrange, N]ewton, and Schwarz: Combinations and Permutations, note = Plenary talk to joint session of the 1999 SIAM Annual Meeting and 1999 SIAM Conference on Optimization, address = Atlanta.*

[6] D. E. Keyes and W. D. Gropp, *A comparison of domain decomposition techniques for elliptic partial differential equations and their parallel implementation*, SIAM Journal on Scientific and Statistical Computing, 8 (1987), pp. S166–S202.

Bulletin

Don't forget to register for the first SIAM Conference on Computational Science and Engineering, Sept 21-24 in Washington DC. It promises to be a great conference and there are several good minisymposium and contributed sessions on optimization in the program. Including the ones mentioned in the first article of this newsletter, there are also sessions on Numerical Optimization in Engineering, Optimal Control and Shape Optimization, Large Scale Nonlinear Programming: Algorithms and Applications, Optimizing Matrix Stability, Optimizing under Uncertainty, and Unstructured Mesh Generation and Optimization.

Comments from the Chair and Editor

We would like to thank Natalia Alexandrov for agreeing to serve as the guest editor for this issue of the SIAG/OPT Views-and-News. She has compiled a very nice set of articles on engineering optimization.

Our intention is to continue to produce at least one newsletter a year that is devoted to a specific area of optimization. We would like to invite others to submit ideas for special issues of the newsletter. If you have a particular area of optimization that you think would be of interest to the general optimization audience please contact either Tom Coleman or myself with the topic and a list of suggested contributors.

Thomas F. Coleman, SIAG/OPT Chair
Computer Science
Cornell University
Ithaca, NY 14853
coleman@tc.cornell.edu

Juan C. Meza, Editor
Sandia National Laboratories
P.O. Box 969, MS 9217
Livermore, CA 94551
meza@ca.sandia.gov